DOI: 10.1111/cgf.12627
Eurographics Conference on Visualization (EuroVis) 2015
H. Carr, K.-L. Ma, and G. Santucci
(Guest Editors)

Volume 34 (2015), Number 3

Visualization of Particle-based Data with Transparency and Ambient Occlusion

Joachim Staib, Sebastian Grottel and Stefan Gumhold

Chair for Computer Graphics & Visualization, TU-Dresden, Germany

Abstract

Particle-based simulation techniques, like the discrete element method or molecular dynamics, are widely used in many research fields. In real-time explorative visualization it is common to render the resulting data using opaque spherical glyphs with local lighting only. Due to massive overlaps, however, inner structures of the data are often occluded rendering visual analysis impossible. Furthermore, local lighting is not sufficient as several important features like complex shapes, holes, rifts or filaments cannot be perceived well.

To address both problems we present a new technique that jointly supports transparency and ambient occlusion in a consistent illumination model. Our approach is based on the emission-absorption model of volume rendering. We provide analytic solutions to the volume rendering integral for several density distributions within a spherical glyph. Compared to constant transparency our approach preserves the three-dimensional impression of the glyphs much better. We approximate ambient illumination with a fast hierarchical voxel cone-tracing approach, which builds on a new real-time voxelization of the particle data.

Our implementation achieves interactive frame rates for millions of static or dynamic particles without any pre-processing. We illustrate the merits of our method on real-world data sets gaining several new insights.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Display Algorithms I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Raytracing

1. Introduction

Numeric simulations are well-established in a variety of research areas, like thermodynamics or molecular biology. Many simulation techniques build on particle-based models for the subject to study, for example atoms or grains. Important steps for data analysis are visualization and interactive exploration. A variety of solutions exist for visualization that provide different metaphors and abstraction levels. One common visualization is to render the particles as opaque spherical glyphs with local lighting. While this simple approach has a low computation time, even for particle counts in the order of millions, many of the spheres are occluded. Especially inner structures that can provide important insight, are hidden. Furthermore, basic lighting models are insufficient and hinder the perception of more complex scene structures like holes, rifts or filaments, as no visual clues for these structures or their depths are conveyed.

To reduce occlusion, most tools for particle based visualizations allow to define opacity values, thus rendering the spheres with a transparency. However, this naive approach is prone to disturb scene understanding, since it does not preserve the volumetric shape of the spheres. In order to improve the depth and shape perception within a scene, ambient occlusion, originally introduced by Zhukov et al. [ZIK98], has become a popular method to approximate global illumination effects with comparably low computational costs.

In this work, we present an illumination model that jointly supports transparency and ambient occlusion for sphere glyphs. Our approach builds on the emission-absorption model of volume rendering. The volumetric shape impression is not only preserved but can be further emphasized through the support of several density distribution functions within a spherical glyph. Ambient occlusion with color information is implemented by approximating the ambient illumination with a fast hierarchical voxel cone-tracing approach. However, we do not resort to classical volume rendering, by

resampling our particle data, since does not meet the expectations of application domain researchers. Instead, we explicitly keep sphere glyph rendering as foundation of our approach.

Our main contributions are:

- A consistent illumination model with joint support for transparency and consistent ambient occlusion
- Closed form solutions of the volume rendering integral for sphere glyphs with a variety of density functions with linear transfer functions.

The remainder of the paper is structured as follows: In section 2 we review related work on visualization of particle-based data, different approaches and uses of transparency and ambient occlusion. Section 3 describes our illumination model. In section 4 the density and transparency model is presented. Section 5 details our implementation and GPU optimizations. We applied our method to real world data, show the results in section 6, followed by a discussion on merits and drawbacks of our approach in section 7. Our conclusion and ideas for future work are presented in section 8.

2. Related Work

2.1. Particle visualization

A variety of tools and frameworks exist for visualizing particle data, especially in the domain of molecular biology. BallView [MHLK06] provides a software library and application for various visualizations, including sphere glyphs. Other such renderers include PyMol [Sch10], widely used in the bio-chemistry community or VMD [HDS96]. All tools incorporate a rich set of rendering techniques and visualization methods, e.g. ribbon or surface representations. Further tools and libraries focus more on performance and serve as a foundation for visualization research. This includes VTK [SML96] and Paraview, VisIt [CBW*12] and SCIRun [Par99]. A more detailed review on tools and frameworks for particle visualization can be found in [GKM*14].

Most of these tools are able to visualize point-based data through sphere glyphs. Since spheres have a very simple parametric form they are especially well suited for ray casting algorithms. Gumhold [Gum03] presents a technique for fast ray casting of ellipsoids on the GPU. Klein and Ertl [KE04] present a similar approach, only differing in the geometry used as canvas for the fragment shader. Grottel et al. [GRDE10] further speed up rendering for time-dependent interactive particle data by occlusion prediction from previous frames. Lindow et al. [LBH12] and Falk et al. [FKE13] utilize instancing for repetitive structures within the data and grid based data structures to reach interactive performance ($>$ 3 FPS) for several billion atoms. Our method also uses GPU ray casting, and will be detailed in section 5.2.

2.2. Transparency

Using transparency in computer graphics is well-established. However, it comes with a price on two fronts: First, it is computationally more expensive, as fragments generated during rasterization of the scene need to be blended in order. A variety of methods to solve this problem exist, from sorting the scene elements, like in our approach, to methods that allow for order independent transparency, like depth peeling [Eve01] or by using per-pixel linked lists in modern GPU buffers with arbitrary read and write access or atomics (cf. [KLZ14]). The latter methods are either approximative, e.g. limited in the number of overlapping layers, or have an additional computational overhead, e.g. through the necessity of synchronization points and additional buffers during rendering. This makes them unsuited when possibly hundreds of transparent layers overlap.

The second challenge of transparency is the aggravation of scene understanding, since depth cues, crucial for human perception [ROP11], are reduced [TM04]. Many approaches exist to alleviate this problem, ranging from additional textures for transparent surfaces [IFP97] to more complex effects like stroking transparent surfaces or brightening colors that are partly occluded [DWE02]. Such approaches are designed for large surfaces, like iso surfaces, and will not work for fragmented surfaces created in particle-based visualization.

In scientific visualization, transparency plays an important role, e.g. in volume visualization, it is an integral part of the absorption transfer function and an established method to render multiple iso surfaces [KLH05]. Most tools for particle visualization like VMD [HDS96] or PyMol [Sch10] treat transparency as an additional material parameter, like the color, ignoring the ramifications of their use. A small number of advanced approaches exists in other fields, for example in vector field visualization [FW08] or view-dependent transparency in line visualization [GRT13]. Carnecky et al. [CFM*13] present an approach to adjust transparency based on X- and T-junction, i.e. visual structural elements of surfaces. The accompanying user study shows that their transparency models performs better than simpler ones.

2.3. Ambient Occlusion

Ambient occlusion, as described by Zhukov et al. [ZIK98] is a fast approximation of global illumination effects. It calculates how much a point on a surface is occluded by its surrounding and attenuates the amount of incoming light, thus roughly modelling missing secondary light. Lindemann et al. [LR11] conduct a comprehensive user study on the influence of global illumination, including ambient occlusion, on perception. They conclude, that costly global illumination delivers by far the best results and that local lighting approaches perform better when combined with ambient occlusion.

Depending on the quality of the approximation and the underlying data, a variety of approaches has been developed.

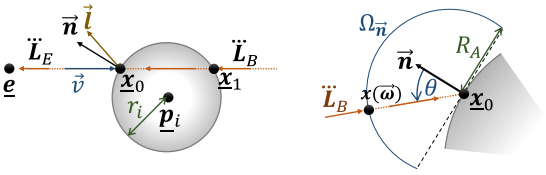


Figure 1: *Notation of our illumination model. Left: surface reflection and volume illumination of particle i. Right: ambient occlusion and illumination at first hit* $\underline{x}_0$.

A technique that evaluates ambient occlusion terms in object space is presented by Pharr et al. [PG04]. Since it requires precomputation of the ambient factors, it is not suited for dynamic data. Hoberock and Jia [HJ07] describe an object-space ambient occlusion method for polygonal data that does not require precomputation. They place disk elements to approximate the scene geometry. While being comparably fast, this method leads to severe artifacts, when the disks over- or undersample the geometry. Crassin et al. [CNS*11] present a technique to efficiently compute global illumination effects, including object space ambient occlusion, for polygonal scenes by generating a lower resolution sparse voxel grid. Their method allows for the calculation of the local obscurance of a scene point by approximate tracing of a low number of cones via a process denoted as voxel cone tracing. Our implementation is comparable with this technique, as will be shown in section 5. In 2007, Mittring et al. [Mit07] present an efficient calculation of screen space ambient occlusion, which is based on sampling the surrounding of a pixel in screen space with simple depth comparisons. However, this approach is not applicable for multiple layers of semi-transparent surfaces, which is the case in our method.

In the field of scientific visualization, many applications of ambient occlusion exist for different scenarios and data types, most notably in the field of volume rendering. The survey [JSYR14] provides an overview on global illumination techniques for volume rendering. For example, Hernell et al. [HLY10] present a volume rendering approach incorporating ambient occlusion. Ambient occlusion is also used for line rendering by Eichelbaum et al. [EHS13]. In the same year, they adopt this approach for point rendering [ESH13]. Ambient occlusion for particle visualization is used by Tarini et al. [TCM06] for their molecular visualization software Qutemol. The core idea of their approach is to generate shadow-map like buffers for a limited set of directions. This technique does not scale for a large number of spheres as the complete scene has to be rendered multiple times. Furthermore, it shares common problems of texture based shadow-mapping approaches, like aliasing or undersampling. Grottel et al. [GKSE12] present an ambient occlusion technique for fully opaque particle visualization based on a coarse voxelization of the scene, which has very low computational cost. Our method shares the idea of a volume representation of the scene. The backside of their approach are under-sampling artifacts and the limited range of the neighborhood considered, due to their very limited sampling schema. Our approach uses a more sophisticated sampling of the scene volume and creates higher visual quality. We show a qualitative comparison of their method, Qutemol and our approach in section 6.

3. Illumination Model

We designed our particle visualization model aiming for an approach which is informative, intuitive, interactive and easy to use. To make the approach informative we support color and opacity mapping of particle attributes to visual attributes. Reduction of the opacity allows to better view the interior parts of the data set. Formally, we define that each particle i provides an RGB color $\ddot{c}_i$ and an opacity Θ_i.

To make the approach intuitive we developed a physically motivated illumination model. We combine volumetric and surface illumination for each particle locally. The overall structure of the data set is emphasized by a mixture of ambient occlusion and ambient illumination. We simplified physically based illumination where it hinders understanding. For example, we ignore deflection of light rays during refraction and reduced the combination of reflected and transmitted light at surfaces to a simple mixture, instead of more complex approaches, like using Fresnel factors.

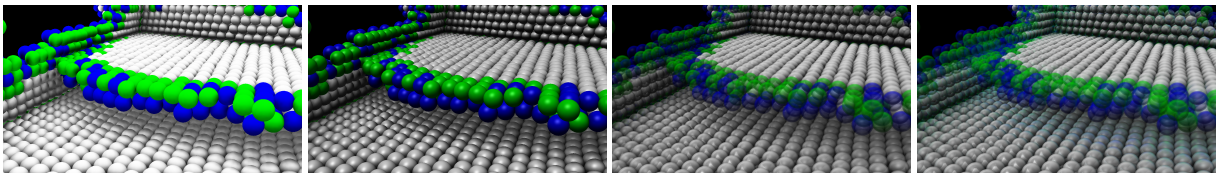
To make the approach interactive we use ray-casting based splatting of the particles. We analytically solve the volume integral along the ray in the interior of the particles. Our approach for computing ambient terms is heavily GPU based and uses volume rasterization and voxel cone tracing.

To make the approach easy to use we put significant effort in reducing the number of visualization parameters. These are two mixing parameters α_{RV} and α_A that trade off between surface reflection and volume illumination and between ambient occlusion and ambient illumination.

Our illumination model is a combination of different parts, as presented in fig. 2. In a) only ambient occlusion is used. We add a surface reflection model in b) with lighting from a directional light source. To evaluate the particle transparency correctly, we add the volumetric emission absorption model to the interior of the particle, as seen in c). This provides a consistently varying opacity over the particle footprint. We allow the user to mix the surface reflection and volume illumination models with $\alpha_{RV} \in [0,1]$, which reflects the user's preferences. To provide more information about the neighboring particles, we also compute a colored ambient illumination term that is added, as shown in d).

We denote radiances with a capital letter $\ddot{L}$ and they are represented as RGB vector. We denote the local illumination $\ddot{L}_{RV}$, the opacity from volume rendering Θ, the ambient occlusion term A_O and ambient illumination $\ddot{L}_A$.

Figure 1 illustrates the most important quantities of our



a) ambient occlusion only b) with surface reflection c) with volumetric model d) with ambient illumination

Figure 2: *The different parts of our illumination model.*

illumination model at one particle. The geometry for each particle glyph i is defined by its center $\underline{\boldsymbol{p}}_i$ and radius r_i. The interior is described by a density function $\rho(t)$ for points t on the viewing ray, defined by a view point $\underline{\boldsymbol{e}}$ and the ray direction $\vec{\boldsymbol{v}}$ for a pixel as $\underline{\boldsymbol{l}}(t) = \underline{\boldsymbol{e}} + t \cdot \vec{\boldsymbol{v}}$.

3.1. Local Particle Illumination

As surface reflection model we calculate the color $\ddot{\boldsymbol{L}}_R(\underline{\boldsymbol{x}}_0)$ using the diffuse term and specular term according to the Blinn-Phong model (omitting the ambient term). We set the diffuse coefficient to the material color $\ddot{\boldsymbol{c}}_i$ and the specular to white.

Based on the density $\rho(t)$ we analytically evaluate the volume rendering integral from the entry point $\underline{\boldsymbol{x}}_0$ to the exit point $\underline{\boldsymbol{x}}_1$, which lie at t_n and t_f on the viewing ray $\underline{\boldsymbol{l}}(t)$. By setting the transfer functions for emission $\ddot{\boldsymbol{\varepsilon}}(\rho)$ and absorption $\sigma(\rho)$ to be linear, the emission color equals the material color $\ddot{\boldsymbol{c}}_i$ and the transparency is $\tau(t_n, t_f)$.

The background illumination $\ddot{\boldsymbol{L}}_B$ is reduced inside the particle to $\tau(t_n, t_f)\ddot{\boldsymbol{L}}_B$. Furthermore, the transparency is related to the opacity via

$$\Theta(t_n, t_f) = 1 - \tau(t_n, t_f) \,. \tag{1}$$

In the same way transparency can be mapped to the surface reflection model. We combine both approaches at $\underline{\boldsymbol{x}}_0$ via α_{RV}

$$\ddot{\boldsymbol{L}}_{RV}(\underline{\boldsymbol{x}}_0) = \Theta(t_n, t_f)\left((1-\alpha_{RV}) \cdot \ddot{\boldsymbol{L}}_R(\underline{\boldsymbol{x}}_0) + \alpha_{RV} \cdot \ddot{\boldsymbol{c}}_i\right) . \tag{2}$$

The volumetric part is thus comparable to the ambient term in traditional surface reflection models.

From the solutions of the rendering integral, we can compute per particle the maximum value $\Theta_{\max}$ for all possible $\Theta(t_n, t_f)$. This allows to scale the transfer function $\sigma(\rho)$ based on the condition that $\Theta_{\max}$ should be equal to the particle opacity parameter Θ_i. This way, we do not need any new parameters for the volumetric model.

3.2. Ambient Occlusion

Finally, we incorporate the ambient occlusion and ambient illumination terms. We start with the ambient occlusion setting from surface rendering illustrated in Fig. 1 b). At the front hit $\underline{\boldsymbol{x}}_0$ with normal $\vec{\boldsymbol{n}}$ the hemisphere $\Omega_{\vec{\boldsymbol{n}}}$ is extended to the radius R_A.

The basic version of ambient occlusion averages the visibility $V(\underline{\boldsymbol{x}}_0, \underline{\boldsymbol{x}}(\vec{\boldsymbol{\omega}})) \in \{0,1\}$ of the points $\underline{\boldsymbol{x}}(\vec{\boldsymbol{\omega}})$ on the hemisphere of radius R_A with respect to $\underline{\boldsymbol{x}}_0$ over all directions $\vec{\boldsymbol{\omega}}$ of the hemisphere. The contributions from different directions $\vec{\boldsymbol{\omega}}$ are furthermore weighted by the cosine term $\cos\theta = \langle \vec{\boldsymbol{n}}, \vec{\boldsymbol{\omega}} \rangle$:

$$A_{SO}(\underline{\boldsymbol{x}}_0, \vec{\boldsymbol{n}}) = \frac{1}{\pi} \int_{\Omega(\vec{\boldsymbol{n}})} V(\underline{\boldsymbol{x}}_0, \underline{\boldsymbol{x}}(\vec{\boldsymbol{\omega}})) \, \langle \vec{\boldsymbol{n}}, \vec{\boldsymbol{\omega}} \rangle \, d\vec{\boldsymbol{\omega}} \,. \tag{3}$$

The cosine weight models a diffuse reflection model. Assuming a constant background illumination $\ddot{\boldsymbol{L}}_B$, one can compute an ambient radiance for a diffuse surface of color $\ddot{\boldsymbol{c}}_i$ independent of the outgoing direction as

$$\ddot{\boldsymbol{L}}_{SA} = \ddot{\boldsymbol{c}}_i \otimes \left(A_{SO} \cdot \ddot{\boldsymbol{L}}_B\right) \,. \tag{4}$$

A_{SO} describes the fraction of background light that reaches $\underline{\boldsymbol{x}}_0$ transformed with the cosine term to the coordinate system of the surface.

We generalize this approach to transparent particles by replacing the visibility term with the integral $\tau_A(\vec{\boldsymbol{\omega}})$ along a ray from $\underline{\boldsymbol{x}}_0$ to $\underline{\boldsymbol{x}}(\vec{\boldsymbol{\omega}})$.

$$A_O(\underline{\boldsymbol{x}}_0, \vec{\boldsymbol{n}}) = \frac{1}{\pi} \int_{\Omega(\vec{\boldsymbol{n}})} \tau_A(\vec{\boldsymbol{\omega}}) \, \langle \vec{\boldsymbol{n}}, \vec{\boldsymbol{\omega}} \rangle \, d\vec{\boldsymbol{\omega}} \,. \tag{5}$$

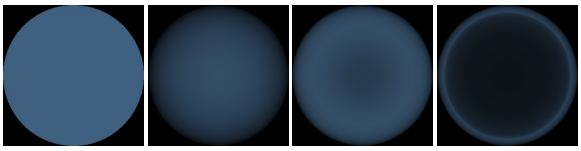
Let $\Theta_A(\vec{\boldsymbol{\omega}}) = 1 - \tau_A(\vec{\boldsymbol{\omega}})$ be the opacity and $\ddot{\boldsymbol{L}}_A(\vec{\boldsymbol{\omega}})$ the volume integral along the same rays. From these we compute the average incoming ambient illumination $\ddot{\boldsymbol{L}}_A$ as

$$\ddot{\boldsymbol{L}}_A = \frac{1}{\pi} \int_{\Omega(\vec{\boldsymbol{n}})} \ddot{\boldsymbol{L}}_A(\vec{\boldsymbol{\omega}}) \, \langle \vec{\boldsymbol{n}}, \vec{\boldsymbol{\omega}} \rangle \, d\vec{\boldsymbol{\omega}} \,. \tag{6}$$

For ambient occlusion in volume rendering [HLY10] used a similar approach, but integrated over a complete sphere, instead of the hemisphere $\Omega(\vec{\boldsymbol{n}})$, thus, losing directional information of the illumination. Note that $\ddot{\boldsymbol{L}}_A(\vec{\boldsymbol{\omega}})$ contains transfer functions for absorption τ_A and emission $\ddot{\boldsymbol{\varepsilon}}_A$, which can be adjusted independently. For example, $\ddot{\boldsymbol{\varepsilon}}_A$ can be used to reduce the emitted radiance for nearly opaque particles (cf. Fig. 2 d)).

We use the ambient occlusion term A_O to modulate the result $\ddot{\boldsymbol{L}}_{RV}$ of surface reflection and volumetric emission. The final formula for the radiance emitted to the eye is

$$\ddot{\boldsymbol{L}}_E = A_O \ddot{\boldsymbol{L}}_{RV} + \alpha_A \ddot{\boldsymbol{L}}_A \,, \tag{7}$$



a) b) c) d)

Figure 3: *Different settings of our volumetric model. a) naive opacity, b) with a constant density function, c) L_2^2 norm density function, d) constant density with 10% border*

and incorporates the ambient illumination as an additive term that can be mixed in and out with the parameter α_A (cf. Fig. 2 d)).

4. Sphere Density Model

For the volumetric lighting $\ddot{\boldsymbol{L}}_V$ at a surface point we explicitly evaluate the volume rendering integral. We assume the spheres to be filled with light emitting and absorbing gas. By careful choice of the density functions for the interior of each sphere, a variety of opacity profiles can be defined. Fig. 3 illustrates the results for different density functions.

Fig. 3 b) shows the opacity of a sphere with constant density. While the spherical shape is much better retained as in the naive approach (cf. Fig. 3 a)), it results in a less perceivable border of the glyphs, as the length of the light ray inside the volume decreases towards zero. This counters our efforts to preserve the shape impression of the sphere. To circumvent this problem, we can compute the density inside the sphere by the L_2^2 norm to the glyph's center (cf. Fig. 3 c)). This de-emphasizes the center and further improves the visibility of the actual border. Details are presented in subsection 4.2. As a second addition, we can explicitly remove a smaller concentric spherical sub-volume, resulting in a hollow sphere, as shown in Fig. 3 d) and further elaborated in subsection 4.3. We simplify the computations using linear transfer functions for emission $\ddot{\boldsymbol{\varepsilon}}(\rho)$ and absorption $\sigma(\rho)$.

As stated before, the maximum opacity value $\Theta_{\max}$, given for each particle, defines the opacity at the most opaque point of the projected sphere glyph. The final color for emission and absorption, but omitting the translucent background, is defined as

$$\ddot{\boldsymbol{L}}_V = \int_{t_n}^{t_f} \tau(t_n,t) \cdot \ddot{\boldsymbol{\varepsilon}}(\rho(t))dt \tag{8}$$

with the transparency function

$$\tau(t_0,t_1) = e^{-\int_{t_0}^{t_1} \sigma(\rho(\tilde{t}))d\tilde{t}}, \tag{9}$$

where $\rho(t)$ denotes one of our density distributions along the viewing ray $\underline{\boldsymbol{l}}(t)$.

Let $\sigma(\rho) = \lambda\rho$ be the linear absorption and $\ddot{\boldsymbol{\varepsilon}}(\rho) = \lambda\ddot{\boldsymbol{c}}_i \cdot \rho$ the linear emission. The linear factor λ will be used for opacity adjustment. $\tau(t_0,t_1)$ then simplifies to

$$\tau(t_0,t_1) = e^{-\lambda P(t_1)} \cdot e^{\lambda P(t_0)}$$

where $P(t)$ denotes the integral of $\rho(t)$. Inserted into Eq. (8) the volume light intensity becomes:

$$\ddot{\boldsymbol{L}}_V = \lambda\ddot{\boldsymbol{c}}_i e^{-\lambda P(t_f)} \int_{t_n}^{t_f} e^{\lambda P(t)} \rho(t)dt \; . \tag{10}$$

By using the substitution rule, Eq. (10) further simplifies to

$$\ddot{\boldsymbol{L}}_V = \lambda\ddot{\boldsymbol{c}}_i e^{-\lambda P(t_f)} \int_{P(t_n)}^{P(t_f)} e^{\lambda t} dt$$

$$\ddot{\boldsymbol{L}}_V = \ddot{\boldsymbol{c}}_i \Theta(t_n,t_f) \tag{11}$$

with

$$\Theta(t_n,t_f) = 1 - e^{-\lambda(P(t_f)-P(t_n))} \tag{12}$$

being the opacity function. This allows the volume rendering integral to be analytically evaluated for any density function $\rho(t)$ that is integrable once.

4.1. Constant Density Function

The constant density function is given trivially as

$$\rho_{cr}(t) = 1 \; . \tag{13}$$

Inserted into Eq. (12), the opacity function reads:

$$\Theta_c(t_n,t_f) = 1 - e^{-\lambda(t_f-t_n)} \; . \tag{14}$$

It has a maximum value when the length of the light ray segment in the inner of the sphere equals twice the radius r_i, i.e. when it crosses the center. This maximum opacity is controlled by the particles parameter $\Theta_{\max}$. Setting Eq. (14) to $\Theta_{\max}$ and solving for λ yields:

$$\lambda = -\frac{1}{2r}\ln(1-\Theta_{\max}) \; . \tag{15}$$

When $\Theta_{\max}$ is 0, the paramter λ will also become 0, leading to a fully transparent sphere. Setting $\Theta_{\max}$ to 1, λ becomes ∞, obtaining a fully opaque sphere.

4.2. L_2^2-Distance to Sphere Center Density Function

This density model uses the L_2^2 distance of each point to the sphere center as density value.

Let the sphere center be at the origin, then the density function reads

$$\begin{aligned}\rho_{l2}(t) &= \langle \underline{\boldsymbol{e}} + t\vec{\boldsymbol{v}}, \underline{\boldsymbol{e}} + t\vec{\boldsymbol{v}} \rangle \\ &= t^2 \cdot \langle \vec{\boldsymbol{v}}, \vec{\boldsymbol{v}} \rangle + t \cdot 2 \langle \underline{\boldsymbol{e}}, \vec{\boldsymbol{v}} \rangle + \langle \underline{\boldsymbol{e}}, \underline{\boldsymbol{e}} \rangle \; .\end{aligned}$$

Its integral is

$$P_{l2}(t) = \int \rho_{l2}(t)dt = \left(\frac{1}{3}\langle\vec{v},\vec{v}\rangle t^3 + \langle\underline{e},\vec{v}\rangle t^2 + \langle\underline{e},\underline{e}\rangle t\right) + C\,. \quad (16)$$

Inserting Eq. (16) into Eq. (12) the final opacity is given by:

$$\Theta_{l2}(t_n,t_f) = 1 - e^{-\lambda\left(\frac{1}{3}\langle\vec{v},\vec{v}\rangle(t_f^3-t_n^3)+\langle\underline{e},\vec{v}\rangle(t_f^2-t_n^2)+\langle\underline{e},\underline{e}\rangle(t_f-t_n)\right)}\,. \quad (17)$$

The ray with maximum opacity is not as obvious as with the constant density. It is obtained by reparametrizing $\Theta_{l2}(t_n,t_f)$ as a function of the distance to the sphere center and calculating its maximum. Its maximum value is $\lambda\frac{2\sqrt{2}}{3}r_i^3$ when the distance equals $\frac{r_i}{\sqrt{2}}$, which yields

$$\Theta_{\max} = 1 - e^{-\lambda\frac{2\sqrt{2}}{3}r_i^3} \quad (18)$$

and

$$\lambda = -\frac{3}{2\sqrt{2}r_i^3}\ln(1-\Theta_{\max})\,. \quad (19)$$

4.3. Hollow Sphere Density Models

The presented mathematical framework evaluates the opacity as a function of the ray entry value t_n and the exit value t_f. Through a straight-forward extension, hollow spheres are supported, i.e. spheres that follow one of the aforementioned density models, but enclose a smaller empty sphere with a radius $r_s < r$. Its opacity is determined by first calculating the intersection points of the inner sphere t_{ns} and t_{fs}. The opacity values for the intervals $[t_n,t_{ns}]$ and $[t_{fs},t_f]$ are computed. The final opacity is obtained by compositing.

5. GPU-Based Implementation

The particles list is transferred to GPU memory into a vertex buffer object. We also allocate an index buffer used to sort the particle data view dependently. In each frame, we update the indices by depth sorting if the view point changes. We use a radix sort prefix-sum sort algorithm [HSO07], implemented in compute shaders. This algorithm is especially well suited for our approach as it is parallel in nature and has a linear run time, independent of the data order.

As next step, the particles are voxelized into a RGBA 3D texture, which is the core data structure needed for the evaluation of the ambient occlusion and illumination terms. This operation is only carried out when the data changes. During voxelization, the density of all spheres is preclassified using our ambient illumination transfer function (cf. Sec. 3.2).

The particle buffer, the index buffer and the 3D texture are then used for rendering. For each sphere, a `GL_POINT` with a size that fully contains the projected sphere is generated in the vertex stage. For each fragment, we cast a ray from the camera through the pixel that covers the fragment. If the sphere is hit, the local illumination model, ambient occlusion and ambient illumination are evaluated. The ambient terms are obtained through voxel cone tracing, comparable to the approach by [CNS*11] (cf. Sec. 5.2).

5.1. Scene Voxelization

The spheres are voxelized into a RGBA 3D texture with a typical size of $256\times256\times256$ bound to a frame buffer target. When 3D textures are bound to a frame buffer, the geometry shader allows for each emitted point primitive to select a slice via the OpenGL variable `gl_Layer`. We employ the same vertex buffer that is used for rendering.

The geometry shader performs a slicing of each sphere. Given position and radius, point primitives are generated for each volume slice that intersects the sphere. We set the size of each primitive to match the maximum radius of the slice.

Fragments that do not contribute to the slice are discarded. For all other fragments, the transfer functions are applied and weighted with a factor w modeling the overlap ratio between the sphere and the voxel cell to avoid aliasing effects. If the current voxel cell contains the complete sphere, then w is the fraction of space that the sphere occupies in the cell. If the voxel cell is completely contained in the sphere, then w is 1. Otherwise, the cell contains the sphere partially. In this case, we sample on a regular sub grid with a resolution of $3\times3\times3$ inside the voxel cell and set w to the fraction of hits. When multiple spheres contribute to a voxel cell, their values are summed up by additive blending.

As final step of the voxelization, a mipmap-pyramid is built for the voxel cone tracing by averaging. Utilizing the compute shader for this calculation, the computation time is reduced by a factor of 30, compared to the standard OpenGL method `glGenerateMipmaps`.

5.2. Ray Casting of Spheres

The spheres are rendered in a front-to-back manner. Each sphere is sent as a `GL_POINT` to a vertex shader that sets the point size to fully contain the projected sphere.

In a fragment shader, a ray-sphere intersection with the current viewing ray is evaluated and the entry and exit position t_n and t_f are determined. Then, the local lighting and the volume rendering integral according to Sec. 4 are evaluated.

Finally, the ambient terms is determined. Depending on a user selectable aperture angle, three cones, tightly aligned around the point's normal, are set up. We perform voxel cone tracing in the previously voxelized scene in an iterative manner. For each cone, the sampling starts at a small offset from the surface point to avoid fetching values originating from the current sphere itself. A RGBA value is fetched from the 3D texture that contains the neighborhood in the domain of

the voxel. Its mipmap level corresponding to the cone diameter at that sampling position. Voxel entries between two discrete mipmap levels are quadrilinearely interpolated by the GPU.

Subsequently, we step one diameter step further along the principal cone direction, take a new sample from the mipmap level that corresponds to the new diameter and compose it in a front-to-back manner. This process is repeated until a predefined cone length is reached. The length is typically set to 25% of the scene extend, but can be adjusted by the user. The final fragment color is blended front-to-back into the screen buffer.

Since the spheres need to be sorted anyway, the rendering process can be accelerated by an approximation of early-z termination. We split the point data into a small set of chunks (in the order of 20) along the z-direction and render them successively into the screen buffer. For each chunk, the rendering pipeline is configured to only process fragments where the value of the depth buffer is 0. After one chunk was rendered and blended, the current RGBA color buffer is evaluated and the depth is set to 1 if the alpha value for a pixel exceeds a maximum threshold, thus masking this pixel for fragment operations during following rendering passes.

6. Results

We use six real world data sets, shown in figure 4, for the evaluation of our method. Their sizes in numbers of particles are given in table 1. For all six data sets we created visualizations using our proposed illumination model.

Data set D1 was included mainly for the purpose of performance measurements. We adjusted the overall transparency to the spheres using the constant density function to show the depth complexity and we use ambient occlusion to show local strands of atoms. Data set D2 shows dislocations and stacking faults in a solid block of a nickel aluminum alloy. The planar stacking faults are colored in gray, only slightly transparent, while the context giving linear dislocations are colored in blue and green, depending on an atom neighborhood classification, with high transparency values. Data set D3 shows three liquid droplets colliding and merging. All droplets are of the same Lennard-Jones material model. The different colors indicate the originating droplet for each atom. We compute the opacity based on the local density, i.e. the number of neighbors for each atom within a given radius for each droplet separately. We de-emphasize the vapor phase and the liquid phase by setting low opacity for low and high density values. Particles in medium density areas are highlighted through high opacity. See the discussion below for a rational on this approach. Data set D4 shows a bulge in a laser ablation simulation in the moment of ripping apart. The opacity was also computed by a local density, simply mapping high density to low opacity and low density to high opacity. Data set D5 shows a bursting layer of liquid in vacuum. We used a similar opacity computation as for D3 with the only difference that D5 consists of only one type of molecules. Data set D6 shows a laser ablation simulation and is with 60 million atoms the largest data set in our evaluation. We used the local density as basis for opacity and color. The bulk material is visually removed by using full transparency for atoms within the respective density interval. Atoms of lower density are colored blue, of higher density red to yellow, with increasing opacity in both cases.

Table 1: *The six data sets used for evaluation*

data set	description	# particles
D1	Protein 1UUN	2,758
D2	NiAl dislocations	50,547
D3	Three colliding droplets	79,509
D4	Laser ablation bulge	509,423
D5	Bursting liquid layer	2,000,000
D6	Laser ablation shock wave	60,000,000

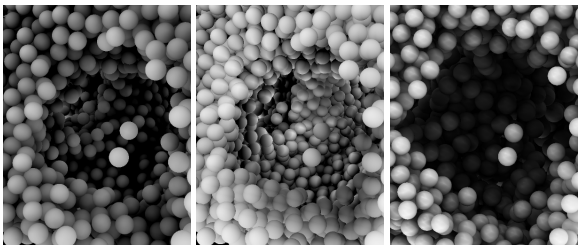


Figure 5: *Comparison of ambient occlusion methods with a close-up of a small portion of dataset D5. Left: our method, center: ambient occlusion from [GKSE12], right: ambient occlusion from QuteMol [TCM06].*

The test system for our performance measurements was an Intel Core i7-3770 CPU (4×3.40) GHz and an Nvidia GeForce GTX 660. The render window had a size of 1024×768 pixels. The resolution of the 3d texture used for voxelization was $256 \times 256 \times 256$ for all data sets. The overall performance numbers in *frames per second* are given in table 2. The results show interactive frame rates for particle counts up to several million particles (cf. data set D5). Our

Table 2: *Performance of our method in* frames per second. Early-z *denotes the rendering mode utilizing our optimization as shown in subsection 5.2. Upload, voxelization and sorting is performed in each frame. For comparison, the results for a renderer that only evaluates* local lighting *is shown.*

data set	early-z	no early-z	local lighting
D1	150.44	187.54	1243.00
D2	93.31	97.67	1217.80
D3	57.54	64.19	695.90
D4	27.44	24.67	222.80
D5	8.76	6.62	146.60
D6	0.34	0.32	2.50

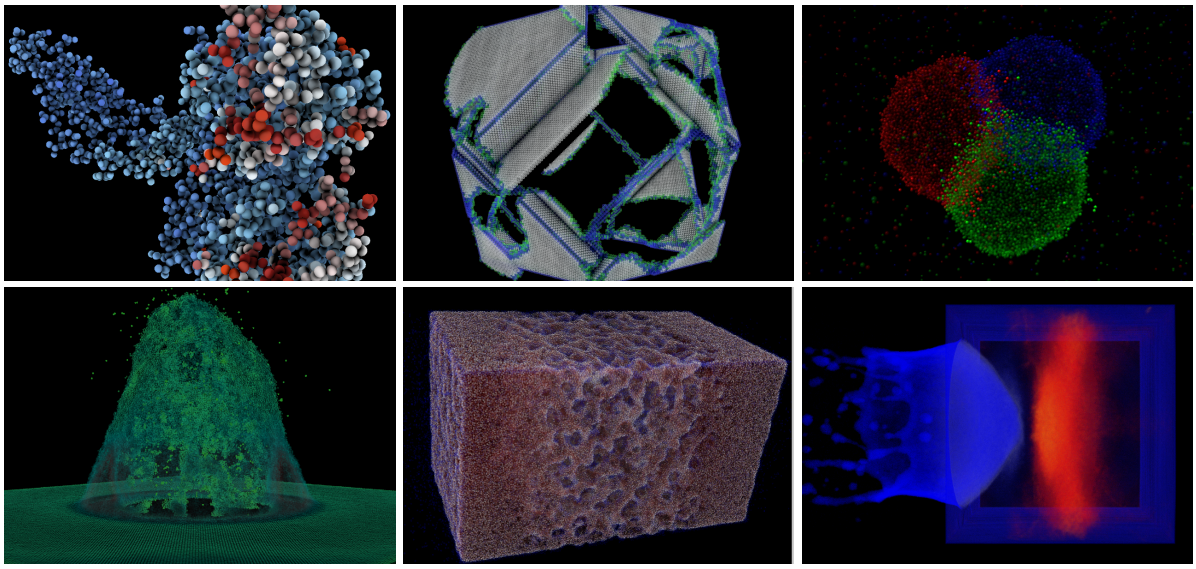


Figure 4: *The six real world data sets used for the evaluation. D1 - D6 from top left to bottom right, according to table 1.*

Table 3: *Performance of different steps in* milliseconds. upload *is the vertex data particle upload,* sort. *is the particle sorting step,* voxel. *is the voxelization step,* render *is the whole fragment shader for the final image generation, and* other *denotes additional overhead. The right sub-table details parts of the* render *stage:* basic *is the fundamental particle ray casting,* transp. *is the evaluation of our volume model, and* amb.occl. *is the computation of the ambient occlusion values.*

data set	upload	sort.	voxel.	render	other	basic	transp.	amb.occl.
D1	0.01	0.26	4.52	1.78	0.05	1.46	0.01	0.31
D2	0.08	0.89	5.07	4.64	0.05	1.72	0.09	2.83
D3	0.42	1.48	5.42	9.35	0.02	2.62	0.10	6.63
D4	3.17	11.11	13.71	8.49	0.11	3.71	0.12	4.66
D5	8.51	42.37	39.09	23.25	0.96	8.02	0.39	14.84
D6	185.10	1367.84	1048.17	517.39	4.19	198.71	2.08	316.60

approximation of early-z termination starts to pay off for data set D4. For smaller particle counts, the overhead of multi-pass rendering outweighs the possible performance gain. Of course, the effectiveness of early-z depends on the settings of the transfer functions, i.e. the opacity accumulation, ranging from small performance gains in D5 to a speedup of factor 5 if the spheres are nearly opaque.

We measured additional details of our algorithm, and show the corresponding computation times in *milliseconds* in table 3. In all cases our early-z optimization was activated. For ambient occlusion calculation, 3 cones are traced per fragment, each require 3 to 5 quadrilinear 3d texture fetches. The results show linear scaling behavior with the number of particles. For small data sizes (D1, D2) the fixed overhead dominates. Data set D3 is an exception to the linear scaling in some stages. We believe this is due to its spatial structure and transparency parametrization, which result in write collisions during voxelization and also makes our early-z optimization ineffective.

To judge our implementation of ambient occlusion qualitatively, figure 5 shows different real-time methods used in particle visualization. Our method nicely darkens local cavities and is nearly artifact-free. At the same time, complex structures, like the hole on the left back side of the cavity, are still visible. Due to its very localized sampling scheme, the method by Grottel et al. [GKSE12] does not darken the center of the rift enough. Also, linear interpolation artifacts are visible. The method in QuteMol [TCM06] suffers even stronger from these artifacts and overall overdarkens the inner of the structure, making it impossible to denote details.

7. Discussion

Our implementation is able to retain interactive rendering rates for data sets with millions of particles. These numbers even include our GPU-based voxelization, mipmap-generation and particle depth sorting. The first two computations, however, are normally only necessary once per data time step. The performance of the voxelization depends not only on the number of particles, but also on their position and splatting order. If many particles overlap the same

voxel, like in D3 and D6, synchronization operations are necessary in the geometry shader stage that increase generation time. Calculating the opacity for each fragment only consists of numeric calculations with moderate run-time complexity and does not require any texture lookups. The computation time for ambient occlusion and emission is low but still creates high visual quality images, using the applied approach of voxel cone tracing. Similar to the method presented in [GKSE12] the image quality of the ambient occlusion depends on the resolution of the volume 3d texture. Our approach, however, is less susceptible to this issue, as the voxel cone tracing takes a larger portion of the volume data into account.

In its current state, only data sets where the spheres do not intersect result in artifact-free images, as the compositing of the individually point-based rendered spheres would produce wrong results otherwise. This, for example, prohibits rendering macro-molecules like proteins in the display style of Van der Waals surfaces. For the rendering of the data set D1 we reduced the radii, so that the spheres only touch each other. We plan to address this issue in future work.

To judge the effectiveness of our visualization qualitatively, especially the shape perception and transparency, we retrieved expert feedback from application domain researcher from physics and thermodynamics. For them, one important task is the visual inspection of data for unexpected features. In data set D2 the reduced occlusion using the semi-transparent representation was regarded helpful, especially in later time steps. The transparency helps for the overview but has no real benefit in close-up inspection of smaller portions of the data. For this second aspect, however, the clearly perceivable shape of our hollow sphere model was judged useful. In the detail view, the ambient occlusion visually separates different layers of atoms (cf. Fig. 2), which was highlighted by the domain experts. In the data set D4, the transparency was commented similarly positive for an overview over internal structures, while shading and ambient occlusion help for the shape perception of structural details.

The comments on the visualization of data set D3 were especially positive. The opacity derived from the local density clearly shows the surfaces of the droplets. Because the local density was computed per droplet independently, after merging, the visualization clearly shows the front of the diffusion process taking place inside the combined droplet. This diffusion process is important but usually hard to see. Our visualization of data set D5 was also received well, as the density emphasizes the interface surface between liquid and vacuum and shows the depth complexity of the breakup. Additionally the ambient occlusion helps understanding the structures emerging during the breakup, especially in early time steps. The visualization of data set D6 revealed a higher-order effect of the simulation the domain experts did not expect at first. The atoms with higher local density clearly show the shock wave from the laser impact moving through the material. More importantly, within this shock wave, distinct diagonal lines are visible which were unexpected. These lines result from reflections of the shock wave onto itself at the boundary conditions of the material block.

In summary, particle data visualization, needs to retain the shape impressions of the individual spheres, but benefits from transparency to lessen the occlusion problem and from ambient occlusion to emphasize large scale structures within the data. Our different density models for the sphere glyphs were very well received in different application scenarios, as was the combination with transparency derived from local density. Ambient occlusion was considered helpful to understand the complexity of larger structures and shapes formed by the particles. To the best of our knowledge, our system is the first to incorporate all these aspects together in a consistent, real-time visualization.

8. Conclusion and Future Work

We presented a model for combined ambient occlusion lighting and transparent sphere glyph rendering. The transparent sphere glyphs are based on analytical solution of the emission-absorption volume rendering equation for different density models. Our approach remedies the problems of occlusion and ill shape perception. Our implementation performs at interactive frame rates for millions or particles. We evaluated our method with real world data sets in terms of rendering performance and in terms of visualization effectiveness through application domain expert feedback. The combined application of transparency and ambient occlusion is beneficial for particle-based data set analysis in the presented scenarios.

As future work we plan on expanding our approach to counter current drawbacks. Most prominently, we want to remedy the issue with intersecting spheres. Possible approaches include manual composition with read-write frame buffer objects and introduction of explicit clipping planes for intersections. Similarly, we plan to integrate a limited number of further graphical elements for more visualization scenarios, like cylinders or cones. We also plan to investigate alternative and extended lighting models, like integrating the ambient occlusion terms directly in the volume light model. With optimizations we target for algorithms with better scaling for interactive rendering of even larger data sets.

Acknowledgements

The authors wish to thank Prof. Jadran Vrabez (University of Paderborn) for his valuable feedback on our data set evaluation. We also thank Ludwig Schmutzler for contributing to the supplemental video. This work was partially founded by BMBF Project No. 01IS14014 (ScaDS).

References

[CBW*12] CHILDS H., BRUGGER E., WHITLOCK B. J., MEREDITH J., AHERN S., BONNELL K., MILLER M., WEBER

G. H., HARRISON C., PUGMIRE D., FOGAL T., GARTH C., SANDERSON A., BETHEL E. W., DURANT M., CAMP D., FAVRE J. M., RUBEL O., NAVRATIL P., WHEELER M., SELBY P.: *VisIt: An End-User Tool for Visualization and Analyzing Very Large Data*, 1 ed., vol. 1 of *CRC Computational Science Series*. Taylor and Francis, Boca Raton, 2012, p. 520. 2

[CFM*13] CARNECKY R., FUCHS R., MEHL S., JANG Y., PEIKERT R.: Smart transparency for illustrative visualization of complex flow surfaces. *IEEE Transactions on Visualization and Computer Graphics 19*, 5 (May 2013), 838–851. 2

[CNS*11] CRASSIN C., NEYRET F., SAINZ M., GREEN S., EISEMANN E.: Interactive Indirect Illumination Using Voxel Cone Tracing. 3, 6

[DWE02] DIEPSTRATEN J., WEISKOPF D., ERTL T.: Transparency in interactive technical illustrations. *Computer Graphics Forum 21* (2002), 2002. 2

[EHS13] EICHELBAUM S., HLAWITSCHKA M., SCHEUERMANN G.: Lineao - improved three-dimensional line rendering. *IEEE Transactions on Visualization and Computer Graphics 19*, 3 (Mar. 2013), 433–445. 3

[ESH13] EICHELBAUM S., SCHEUERMANN G., HLAWITSCHKA M.: Pointao - improved ambient occlusion for point-based visualization. In Hlawitschka and Weinkauf [HW13], pp. 13–17. 3

[Eve01] EVERITT C.: Interactive order-independent transparency. *White paper, nVIDIA 2*, 6 (2001), 7. 2

[FKE13] FALK M., KRONE M., ERTL T.: Atomistic visualization of mesoscopic whole-cell simulations using ray-casted instancing. *Computer Graphics Forum 32*, 8 (2013), 195–206. 2

[FW08] FALK M., WEISKOPF D.: Output-sensitive 3d line integral convolution. *IEEE Transactions on Visualization and Computer Graphics 14*, 4 (July 2008), 820–834. 2

[GKM*14] GROTTEL S., KRONE M., MÜLLER C., REINA G., ERTL T.: Megamol – a prototyping framework for particle-based visualization. *Visualization and Computer Graphics, IEEE Transactions on* (2014), (to appear). 2

[GKSE12] GROTTEL S., KRONE M., SCHARNOWSKI K., ERTL T.: Object-Space Ambient Occlusion for Molecular Dynamics. In *Pacific Visualization Symposium (PacificVis), 2012 IEEE* (2012), pp. 209–216. 3, 7, 8, 9

[GRDE10] GROTTEL S., REINA G., DACHSBACHER C., ERTL T.: Coherent culling and shading for large molecular dynamics visualization. In *Computer Graphics Forum (Proceedings of EUROVIS 2010)* (2010), vol. 29, pp. 953–962. 2

[GRT13] GÜNTHER T., RÖSSL C., THEISEL H.: Opacity optimization for 3d line fields. *ACM Trans. Graph. 32*, 4 (July 2013), 120:1–120:8. 2

[Gum03] GUMHOLD S.: Splatting illuminated ellipsoids with depth correction. In *Proceedings of International Workshop on Vision, Modeling, and Visualization* (November 2003), pp. 245–252. 2

[HDS96] HUMPHREY W., DALKE A., SCHULTEN K.: Vmd: visual molecular dynamics. *Journal of molecular graphics 14*, 1 (1996), 33–38. 2

[HJ07] HOBEROCK J., JIA Y.: High-quality ambient occlusion. *GPU gems 3* (2007), 257–274. 3

[HLY10] HERNELL F., LJUNG P., YNNERMAN A.: Local Ambient Occlusion in Direct Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics 16* (2010), 548–559. 3, 4

[HSO07] HARRIS M., SENGUPTA S., OWENS J. D.: Parallel prefix sum (scan) with CUDA. In *GPU Gems 3*, Nguyen H., (Ed.). Addison Wesley, August 2007, ch. 39, pp. 851–876. 6

[HW13] HLAWITSCHKA M., WEINKAUF T. (Eds.):. *EuroVis - Short Papers* (Leipzig, Germany, 2013), Eurographics Association. 10

[IFP97] INTERRANTE V., FUCHS H., PIZER S. M.: Conveying the 3d shape of smoothly curving transparent surfaces via texture. *IEEE Transactions on Visualization and Computer Graphics 3*, 2 (Apr. 1997), 98–117. 2

[JSYR14] JÖNSSON D., SUNDÉN E., YNNERMAN A., ROPINSKI T.: A Survey of Volumetric Illumination Techniques for Interactive Volume Rendering. *Computer Graphics Forum 33*, 1 (2014), 27–51. 3

[KE04] KLEIN T., ERTL T.: Illustrating magnetic field lines using a discrete particle model. In *Proceedings of the Vision, Modeling, and Visualization Conference 2004 (VMV 2004), Stanford, California, USA, November 16-18, 2004* (2004), Girod B., Magnor M. A., Seidel H., (Eds.), Aka GmbH, pp. 387–394. 2

[KLH05] KANODIA R. L., LINSEN L., HAMANN B.: Multiple transparent material-enriched isosurfaces. In *WSCG (Full Papers)* (2005), pp. 23–30. 2

[KLZ14] KNOWLES P., LEACH G., ZAMBETTA F.: Fast sorting for exact oit of complex scenes. *The Visual Computer 30*, 6-8 (2014), 603–613. 2

[LBH12] LINDOW N., BAUM D., HEGE H.-C.: Interactive rendering of materials and biological structures on atomic and nanoscopic scale. *Computer Graphics Forum 31*, 3pt4 (2012), 1325–1334. 2

[LR11] LINDEMANN F., ROPINSKI T.: About the influence of illumination models on image comprehension in direct volume rendering. *Visualization and Computer Graphics, IEEE Transactions on 17*, 12 (Dec 2011), 1922–1931. 2

[MHLK06] MOLL A., HILDEBRANDT A., LENHOF H.-P., KOHLBACHER O.: Ballview: a tool for research and education in molecular modeling. *Bioinformatics 22*, 3 (2006), 365–366. 2

[Mit07] MITTRING M.: Finding next gen: Cryengine 2. In *ACM SIGGRAPH 2007 Courses* (New York, NY, USA, 2007), SIGGRAPH '07, ACM, pp. 97–121. 3

[Par99] PARKER S. G.: *The Scirun Problem Solving Environment and Computational Steering Software System*. PhD thesis, The University of Utah, 1999. AAI9937605. 2

[PG04] PHARR M., GREEN S.: Ambient occlusion. *GPU Gems 1* (2004), 279–292. 3

[ROP11] ROPINSKI T., OELTZE S., PREIM B.: Survey of glyph-based visualization techniques for spatial multivariate medical data. *Computers & Graphics* (March/April 2011). accepted for publication. 2

[Sch10] SCHRÖDINGER, LLC: The PyMOL molecular graphics system. http://pymol.org/, August 2010. 2

[SML96] SCHROEDER W., MARTIN K., LORENSEN B.: *The Visualization Toolkit: An Object-oriented Approach to 3D Graphics*. The Visualization Toolkit: An Object-oriented Approach to 3-D Graphics. Prentice Hall PTR, 1996. 2

[TCM06] TARINI M., CIGNONI P., MONTANI C.: Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1237–1244. 3, 7, 8

[TM04] TORY M., MOLLER T.: Human factors in visualization research. *Visualization and Computer Graphics, IEEE Transactions on 10*, 1 (Jan 2004), 72–84. 2

[ZIK98] ZHUKOV S., IONES A., KRONIN G.: An ambient light illumination model. In *Rendering Techniques* (1998), Drettakis G., Max N. L., (Eds.), Springer, pp. 45–56. 1, 2